TITLE OF THE INVENTION

PLATFORM-INDEPENDENT EXCEPTIONS-BASED METHODS AND SYSTEMS FOR REMOTELY MANAGING NODES WITHIN A COMMUNICATIONS NETWORK

PRIORITY APPLICATION

This application claims the benefit of US Provisional Application No. 60/130,144 filed April 20, 1999, entitled "Platform-Independent Exceptions-Based Communications Network System Management Console," and is incorporated herein by reference.

CROSS REFERENCE TO RELATED APPLICATIONS

The application is related to US Provisional Application No. 60/138,348, filed June 9, 1999, entitled "System and Method for Automated Information Retrieval (AIRS)," and is incorporated herein by reference.

This application is related to US Provisional Application No. 60/156,684, filed September 29, 1999, entitled "System and Method for Remotely Monitoring Hardware and Software Devices (NTDS Management Agents)," and is incorporated herein by reference.

COPYRIGHT NOTIFICATION

A portion of the disclosure of this patent document and its figures contain material that is subject to copyright protection. The copyright owner has no objection to the facsimile reproduction by anyone of the patent document or the patent disclosure, as it appears in the Patent and Trademark Office patent files or records, but otherwise reserves

all copyrights whatsoever.

BACKGROUND OF THE INVENTION

1. Field of the Invention

The present invention relates to a methods and systems for remotely managing exceptions within network nodes, and in particular to a platform-independent exceptions-based system management system that is accessible by any computer platform that supports, for example, a Java application or a Java applet.

2. Background

Many types of systems, such as networks, are used to transport information and data between nodes and users at those nodes. The banking industry, for example, provides one example of a network system in which delivery systems, such as Automated Teller Machines (ATMs) or business servers, are coupled to several banks. The banks are the primary nodes and the delivery systems are secondary nodes. Primary nodes are typically hubs that perform a substantial portion of work, while the secondary nodes are used intermittently. Secondary nodes must interface with a primary node in order to operate. In the banking example, a delivery system, such as an ATM, must access a bank in order to determine if the user has enough money in a particular account before dispensing any money. A large banking system may consist of a plurality of delivery systems that are capable of accessing a plurality of banks.

A telephone system is another example of such a system. Telephone systems may also be comprised of, for example, a plurality of networks, with each network containing

a plurality of nodes. Switches and Private Branch Exchanges (PBXs) are primary nodes and individual telephones are secondary nodes. In order for a telephone user to receive and transmit information, the user must be capable of accessing another telephone via a switch or PBX.

Another example is the cable television systems. The central cable company is the primary node that transmits video data via a network to the individual television converters as the secondary nodes. As in other examples of such systems, data flow is bi-directional -- cable television users transmit program selection data via the system to the central cable company node and the cable company transmits video and audio data to the users' televisions.

In the foregoing examples, system performance may occasionally be disrupted due to exceptions caused by hardware or software problems. As used herein, the term "exception" includes any errors, faults, or problems reported for a communication node, including, but not limited to mechanical problems (e.g., a jammed door or malfunctioning key in an ATM node), software problems (e.g., an error in keeping time of certain transactions), and/or node maintenance (e.g., an ATM that has "low cash" and needs to be serviced). Therefore, these systems must be monitored so that exceptions that occur from time to time within the system can be identified and corrected.

In general, system management systems exist to aid in the management of remote systems. System performance and user satisfaction with system management systems depend on the speed and efficiency with which exceptions can be detected and remedied in these remote systems. Remote systems can be normal user desktops, remote servers

that provide network or database services for other systems, or backend mainframe service providers. The management of these many systems is required to ensure the maximum uptime and capabilities of these systems. Usually, a single (or small number) of management servers support this need, and a dedicated staff of operations personnel is assigned to man the management system and to act on problems in the remote system domain.

These system management systems present remote system-status information on computer displays. Older systems typically used text based displays. Newer systems use modern graphical user interfaces (GUIs). Most of these displays overwhelm the user with too much status, mixing good status with exceptions.

A very common display screen will show green colored squares representing systems with no problems mixed with a few non-green colored squares representing systems in trouble. This type of screen does not allow the user to (1) easily detect exception-events that have recently changed and (2) explain what type of exception-event has occurred. The green squares are unnecessary noise. If a user is managing hundreds, or even thousands of systems, then this type of display becomes very unmanageable. Additionally, current methods of managing system performance often require manual exception data analysis, and are therefore labor intensive and time consuming. For these reasons, current system management systems provide much more information than is useful or necessary to rapidly identify exceptions in an efficient manner.

Thus, a need exists for a methods and systems for remotely accessing network nodes and for remotely managing system exceptions to solve the above problems, as well

as to provide additional flexibility and functionality in identifying and implementing corrective action. A need also exists for an exception-based system management system that is able to collect and accumulate information on the type and frequency of exceptions in a fast, reliable, and efficient manner. There is a further need for a flexible exception-based system management system that is able to: (1) manage exceptions in an efficient manner; (2) accommodate additional nodes as the system expands; (3) operate on many different computer platforms; and (4) operate with many different operating systems.

SUMMARY OF THE INVENTION

To overcome these problems, the present invention provides easy, reliable, and efficient exception-based management methods and systems capable of presenting, isolating, categorizing, and resolving exceptions within network nodes.

In an embodiment of the methods and systems of the present invention, a network provides a front end interface, such as, for example, in the form of a computer software application that provides a template for viewing, selecting, and entering information in order to monitor an exception. For example, instead of manually filling out forms and reports to correct system exceptions, a report called a "trouble ticket" is created in an electronic transfer medium, such as, for example, an interactive web page displayed on a client terminal coupled with a network management system server connected to a network. A user, such as a network administrator, creates the trouble ticket by responding to an exception and entering commands and information to resolve the exception. The trouble ticket is electronically transmitted to the network where the data contained in the trouble ticket is used to correct or to document an exception.

The exception-based system management system is a client terminal that works in conjunction with a network management system server. It gets all the status data it needs to paint the screens by requesting the needed data from the network management system server.

In a preferred embodiment of the present invention, the exception-based system management system is a Java program capable of running on any platform that has a Java Runtime Environment (JRE), such as, for example, JRE version 2.0 which is well known in the art. Today, this includes Windows (98 & NT), various UNIX products (e.g., Solaris, LINUX, HP-UX), and in the near future Macintosh. This embodiment allows the present invention to be platform independent. The Java program is downloaded to the client machine by either 1) accessing a web page on the network management system server and running as a Java applet, or 2) accessing a web page on the network management system server and downloading the Java code to a local file and later running the Java program locally. Although in this embodiment, the exception-based system management system client looks like a web browser, it is not. For the most part, once the Java program is running on the client terminal, no web access is required.

In another embodiment of the present invention, the exception-based system management system supports "user modules" that represent screens displayed on a client terminal and allow a user to view, input, select, and/or transmit data. The user module known as the "Exception" Module shows only systems with exceptions. It displays a list of system names along with icons that represents the highest priority problem on each system. This module also prioritizes the systems based on the severity of exception, with

the highest priority exceptions shown first. When a user has acted on the problem by either taking ownership of the exception or opening a trouble ticket on the exception, the icon color is changed and the priority is dropped. This allows the users to separate the new exceptions from the existing ones. The workflow that is supported by the present invention is known as "exception-based monitoring" and starts from the user module knows as the "Exception" Module.

In other embodiments of the present invention, the methods and systems support the user acting on individual exceptions. A user can zoom into a specific system by clicking on the exception icon that takes the user to a more detailed user module called the "Detail" Module. If a user needs to act on the remote system, the user can issue a command to the system via the "Command" Module. The user can track the exception by creating a trouble ticket, which acts as a scratch pad and allows the user to keep notes on the exception. There is also a module to get a business branch view (i.e., view of the segments or offices within a business) of the network and a "Statistics" module that shows graphs of how many systems are in down or degraded states.

In addition to providing the methods and systems outlined above, the present invention: (1) operates on a multitude of hardware platforms and/or operating systems; (2) provides real-time information on exceptions; (3) suggests corrective action to resolve an exception; (4) utilizes a user-friendly GUI; (5) facilitates direct commands to the node resources to initiate corrective action; and (6) utilizes the network to reach the nodes instead of specialized lines.

In different possible embodiments of the present invention, the methods and

systems may be utilized to perform one or more of the following tasks: (1) provide integration with external systems; (2) provide integration with other internal systems; (3) utilize internally maintained data; (4) provide on-line status information; (5) allow for real-time system modifications; (6) provide detailed reports; and/or (7) utilize state-of-the-art communications technology (e.g., web-based technology).

In another possible embodiment of the present invention, a user may provide, additional types of information that may be utilized in the process to identify, initiate, report, and correct exceptions. This includes initiating automated commands to correct an exception.

In a first illustrative embodiment, a platform-independent method for managing exceptions in at least one network having a plurality of nodes interconnected with communication lines, comprises:

remotely accessing at least one communications network having a plurality of nodes interconnected with communication lines;

remotely storing exception data;

remotely prioritizing said exception data;

remotely monitoring said exception data;

remotely transmitting a corrective response to a destination node, wherein said corrective response is identified by a destination node command; and

remotely monitoring said destination node command associated with said destination node to determine a status of said corrective response.

In this first illustrative embodiment, the method may further comprise:

remotely administering said exception data;

remotely administering destination node command data;

remotely constructing an exceptions commands log;

remotely administering said exceptions command log;

remotely printing said exceptions command log;

remotely constructing a report, wherein said report is a trouble ticket associated with said exception data; and

remotely providing a help mechanism to a user.

In this first illustrative embodiment, the trouble ticket may further comprise said destination node command associated with said exception data and allow for:

remotely storing said trouble ticket;

remotely administering said trouble ticket; and

remotely printing said trouble ticket.

In this first illustrative embodiment, the exception data may further comprise identification of at least one destination node categorized by at least one of the following parameters for said destination node:

node filtering;

device filtering;

message filtering; and

audible alert filtering.

Further, in this first illustrative embodiment, the nodes may comprise one or more of the following:

a plurality of delivery system nodes;

a plurality of secondary system nodes;

automated teller machines;

bank servers;

communication servers; and

financial servers.

Finally, in this first illustrative embodiment, the communications network may be a financial institution's communications network, such as a bank's communication network.

In a second illustrative embodiment, a platform-independent system for managing exceptions in at least one communications network having a plurality of nodes interconnected with communication lines, comprises:

means for remotely accessing at least one communications network having a plurality of nodes interconnected with communication lines;

means for remotely storing exception data;

means for remotely prioritizing said exception data;

means for remotely monitoring said exception data;

means for remotely transmitting a corrective response to a destination node, wherein said corrective response is identified by a destination node command; and

means for remotely monitoring said destination node command associated with said destination node to determine a status of said corrective response.

In this second illustrative embodiment, the system may further comprise:

means for remotely administering said exception data;

means for remotely administering destination node command data;

means for remotely constructing an exceptions commands log;

means for remotely administering said exceptions command log;

means for remotely printing said exceptions command log;

means for remotely constructing a report, wherein said report is a trouble ticket associated with said exception data; and

means for remotely providing a help mechanism to a user.

In this second illustrative embodiment, the trouble ticket may further comprise said destination node command associated with said exception data and allow for:

means for remotely storing said trouble ticket;

means for remotely administering said trouble ticket; and

means for remotely printing said trouble ticket.

In this second illustrative embodiment, the exception data may further comprise identification of at least one destination node categorized by at least one of the following parameters for said destination node:

node filtering;

device filtering;

message filtering; and

audible alert filtering.

Further, in this second illustrative embodiment, the nodes may comprise one or more of the following:

a plurality of delivery system nodes;

a plurality of secondary system nodes;

automated teller machines;

bank servers;

communication servers; and

financial servers.

Finally, in this second illustrative embodiment, the communications network may be a financial institution's communications network, such as a bank's communication network.

In a third illustrative embodiment, a method for detecting, isolating, categorizing, and resolving exceptions within network nodes, comprises:

displaying a user module for viewing, selecting, inputting, and transmitting a request from a user to a network exception-based system management system;

accepting said request upon submission by said user;

transmitting exception data associated with a destination node from said request to said exception-based system management system;

translating said exception data into a corrective action work request;

processing said corrective action work request;

storing results from said corrective action work request; and

sending said results to be displayed by said user interface.

In this third illustrative embodiment, the method may further comprise:

administering said exception data associated with said destination node;

managing said exception data associated with said destination node;

administering said results associated with said destination node;

managing said results associated with said destination node; and

providing a help mechanism to said user.

In this third illustrative embodiment, the corrective action work request comprises an on-line request to monitor at least one of the destination nodes in real-time. The corrective action work request may further comprise a destination node command to initiate a corrective response to at least one of the destination nodes in real-time.

In this third illustrative embodiment, the user interface may comprise at least one of the following user modules selected from a group of user modules comprising:

a login module;

an administration module;

a branch module;

a detail module;

an exception module;

a command module;

a ticket module;

a ticket browser module; and

a status module.

Further, in this third illustrative embodiment, the network nodes may comprise one or more of the following:

a plurality of delivery system nodes;

a plurality of secondary system nodes;

automated teller machines;

bank servers;

communication servers; and

financial servers.

Finally, in this third illustrative embodiment, the communications network may be a financial institution's communications network, such as a bank's communication network.

In a fourth illustrative embodiment, a system for detecting, isolating, categorizing, and resolving exceptions within network nodes, comprises:

means for displaying a user module for viewing, selecting, inputting, and transmitting a request from a user to a network exception-based system management system;

means for accepting said request upon submission by said user;

means for transmitting exception data associated with a destination node from said request to said exception-based system management system;

means for translating said exception data into a corrective action work request; means for processing said corrective action work request;

means for storing results from said corrective action work request; and

means for sending said results to be displayed by said user interface.

In this fourth illustrative embodiment, the system may further comprise:

means for administering said exception data associated with said

destination node;

means for managing said exception data associated with said destination node;

means for administering said results associated with said destination node;

means for managing said results associated with said destination node; and

means for providing a help mechanism to said user.

In this fourth illustrative embodiment, the corrective action work request comprises an on-line request to monitor at least one of the destination nodes in real-time. The corrective action work request may further comprise a destination node command to initiate a corrective response to at least one of the destination nodes in real-time.

In this fourth illustrative embodiment, the user interface may comprise at least one of the following user modules selected from a group of user modules comprising:

a login module;

an administration module;

a branch module;

a detail module;

an exception module;

a command module;

a ticket module;

a ticket browser module; and

a status module.

Further, in this fourth illustrative embodiment, the network nodes may comprise

one or more of the following:

a plurality of delivery system nodes;

a plurality of secondary system nodes;

automated teller machines;

bank servers;

communication servers; and

financial servers.

Finally, in this fourth illustrative embodiment, the communications network may be a financial institution's communications network, such as a bank's communication network.

In a fifth illustrative embodiment, a platform-independent system for managing exceptions in at least one communications network having a plurality of nodes interconnected with communication lines, comprises:

a network exception-based system management system coupled to at least one communications network having a plurality of nodes;

an applet that is sent with a web page to said network exception-based system management system; and

a plurality of client terminals, coupled to said applet via said communications network, for user interaction with said network exception-based system management system.

In this fifth illustrative embodiment, the communications network may further comprise:

memory;

at least one database stored in memory; and

at least one database processor capable of processing data contained in said database.

In this fifth illustrative embodiment, the system may further comprise a request to the network exception-based system management system, wherein the request is communicated to the network exception-based system management system by the user interaction. In addition, the system may further comprise an on-line request to monitor at least one of the nodes associated with an exception in real-time, wherein the request further comprises a destination node command to initiate a corrective response to at least one of the nodes associated with an exception in real-time.

In this fifth illustrative embodiment, user interaction involves a user submitting a request to the exception-based system management system by filling out the pre-formatted web page form presented by the Java applet or the Java application. The pre-formatted web page form serves as an interface and may be served up at a remote terminal located at a different site than the communications network. Furthermore, the pre-formatted web page may be served via a wide area network, the Internet, an Intranet or Extranet, a satellite communications network, and/or other network capable of transmitting data.

Finally, in this fifth illustrative embodiment, the pre-formatted web page is a user module that represent a screen displayed on the client terminal. The user module prompts the user to view, input, select, and/or transmit data. User modules may comprise at least

one of the user modules selected from group consisting of:

a login module;

an administration module;

a branch module;

a detail module;

an exception module;

a command module;

a ticket module;

a ticket browser module; and

a status module.

In an alternate illustrative embodiment, instead of a said applet being sent with said web page, an application may be downloaded from said web page to said network exception-based system management system. This allows the system to be downloaded to a local network or to a local computer.

In addition to providing the methods and systems outlined above, the present invention provides management tools that allow data collected to be mined and utilized for management functions, including, but not limited to, monitoring exceptions, profiling exceptions, tracking corrective actions, customer satisfaction, and related functions. The management tools further include the ability to provide reports relevant to the aforementioned management functions. Furthermore, the methods and systems of the present invention may include a wide variety of features to advantageously accomplish different tasks and/or business objectives related to exceptions management.

Further details on these embodiments, other possible embodiments, and the methods and systems of the present invention are set forth below.

As will be appreciated by those of ordinary skill in the art, the methods and systems of the present invention have wide utility in a number of areas as illustrated by the wide variety of possible uses discussed below.

One possible use of the present invention is to provide an exception-based system management system capable of generating reports and statistics on the type and frequency of exceptions in an efficient manner.

Another possible use of the present invention is to store and maintain data on the following: (1) exceptions; (2) trouble tickets; and (3) corrective action decisions.

Another possible use of the methods and systems of the present invention is to provide an exception-based system management system capable of analyzing data and suggesting corrective action to remedy exceptions.

Another possible use of the present invention is to provide comprehensive analysis tools for categorizing, suggesting, and/or automatically initiating corrective action to an exception.

Another possible use of the methods and systems of the present invention is to provide an exceptions-based system management system that is responsive to exceptions as they arise in the system being monitored.

Another possible use of the present invention is to provide a communications network system for collecting and transmitting a trouble ticket between one or more users and/or one or more businesses.

Another possible use of the present invention is to provide a user with capability to conveniently create a trouble ticket and to conveniently transmit the trouble ticket to one or more users, one or more businesses, and/or one or more business branches at a time when it is convenient to the user to do so.

Another possible use of the present invention is to provide for editing of a trouble ticket.

Another possible use of the present invention is to allow a user to: (1) enter and submit trouble tickets; (2) track the status of all trouble tickets; (3) direct, redirect and queue trouble tickets; (4) modify previously submitted trouble tickets; and (6) use modules and information similar to those used by other users and/or a business.

Another possible use of the methods and systems of the present invention is to provide a flexible exceptions-based system management system that is capable of accommodating changes in the system architecture.

Another possible use of the present invention is to make it easier for businesses to accommodate growth in the number of network nodes.

Another possible use of the methods and systems of the present invention is to provide an exceptions-based system management system that is capable of running on many different hardware platforms and with many different operating systems.

Another possible use of the present invention is to interface and communicate with the network communications system through a variety of electronic mediums, including wireline and wireless technology, such as, for example, WAN, LAN, satellite system, telephone lines, and the like.

Another possible use of the methods and systems of the present invention is to utilize web-based technology and minimize impact on the embedded code running inside the node resources and to utilize the communications network to reach the nodes instead of specialized lines in a reliable and efficient manner.

Another possible use of the methods and systems of the present invention is to provide direct commands to the node resources to initiate corrective action in an efficient manner.

Another possible use of the present invention is to allow for single data entry in order to eliminate the mistakes caused by the re-entry of data by multiple users, and accordingly, to reduce the need for personnel to enter exceptions data, including trouble ticket and corrective action data.

Another possible use of the present invention is to significantly reduce the time required by the overall exceptions management process (i.e., the time it takes to detect, isolate, categorize, and resolve exceptions within network nodes).

Another possible use of the present invention is to minimize the amount of paper work generated by: (1) detecting, isolating, categorizing, and resolving exceptions within network nodes; (2) filling out trouble tickets; and (3) tracking exceptions.

Another possible use of the present invention is to provide an exception-based system management system for a financial institution's communications network.

Another possible use of the present invention is to provide on-line system help to the user.

Another possible use of the present invention is to provide real-time entry, editing,

and review of exception data, of trouble ticket data, and of corrective action decisions.

Another possible use of the present invention is to provide for multiple levels of user access and to facilitate multiple levels of security related to those levels of user access.

Another possible use of the present invention is to secure the source code on the network management system server.

Another possible use of the present invention is to provide a user with access to a variety of optional additional useful administrative features, such as, for example, an overview of the node network, filtering capabilities, registration information, runtime control, and a help mechanism.

Another possible use of the present invention is to allow for node filtering, device filtering, message filtering, and/or audible-alarm filtering.

Another possible use of the present invention is to allow for node filtering based on one or more selected nodes, nodes affiliated with a particular business, and/or nodes affiliated with a particular business branch (i.e., segments or offices within a business).

Another possible use of the present invention is to allow for device filtering based on single nodes, all devices, and/or selected sub-devices (e.g., the cash dispensers of ATMs).

Another possible use of the present invention is to allow for message filtering based on category of exception and/or exception code.

Another possible use of the present invention is to allow for audible-alarm filtering based on category of exception and/or exception code.

Another possible use of the present invention is to have one standardized exceptions management interface regardless of a user's computer system (i.e., the hardware, operating system, and/or other software).

Another possible use of the present invention is to allow a user to store data on a local computer or local network.

Another possible use of the present invention is to provide for the use of similar modules when one or more users and/or one or more businesses are discussing an exception and/or trouble ticket.

Another possible use of the present invention is to permit a user to know immediately whether an exception has been corrected and the method of resolving that exception.

These uses may be accomplished singularly, or in combination, in one or more of the embodiments of the present invention.

Additional uses, objects, advantages, and novel features of the invention will be set forth in part in the detailed description that follows, and in part will become more apparent to those skilled in the art upon examination of the following or upon learning by practice of the invention.

BRIEF DESCRIPTION OF THE FIGURES

Other objects and advantages of the invention will be more clearly understood by reference to the following description taken in connection with the accompanying figures, in which:

Figure 1 illustrates an overview of the network topology in an embodiment of the

methods and systems for remotely managing exceptions within network nodes.

Figures 2A and 2B illustrate software component object models in an embodiment of the methods and systems for remotely managing exceptions within network nodes.

Figure 3 illustrates a "Login" Module that a user views, inputs, selects, and/or transmits login information in an embodiment of the methods and systems for remotely managing exceptions within network nodes

Figures 4A-4H illustrate modules that a user views, selects, inputs, and/or transmits administrative information in an embodiment of the methods and systems for remotely managing exceptions within network nodes.

Figure 5 illustrates a "Branch" Module that a user views, inputs, selects, and/or transmits business branch information in an embodiment of the methods and systems for remotely managing exceptions within network nodes.

Figure 6 illustrates an "Exception" Module that a user views, inputs, selects, and/or transmits exception information in an embodiment of the methods and systems for remotely managing exceptions within network nodes.

Figure 7 illustrates a "Detail" Module that a user views, inputs, selects, and/or transmits detailed exception information in an embodiment of the methods and systems for remotely managing exceptions within network nodes.

Figure 8 illustrates a "Command" Module that a user views, inputs, selects, and/or transmits command information in an embodiment of the methods and systems for remotely managing exceptions within network nodes.

Figures 9A and 9B illustrate ticket modules that a user views, inputs, selects,

and/or transmits trouble ticket information in an embodiment of the methods and systems for remotely managing exceptions within network nodes.

Figure 10 illustrates a "Status" Module that a user views, inputs, selects, and/or transmits status information in an embodiment of the methods and systems for remotely managing exceptions within network nodes.

DESCRIPTION OF ILLUSTRATIVE EMBODIMENTS

The methods and systems of the present invention provides an exception-based system management system capable of detecting, isolating, categorizing, and resolving exceptions within network nodes. Some of the possible benefits of the present invention include: (1) minimizing manual entry of exceptions; (2) minimizing exceptions due to multiple entries; (3) minimizing time between reporting exceptions and initiating corrective action; (4) generating similar sets of exception data to multiple users and/or businesses; (5) maintaining real-time tracking and status; (6) storing data that is accessible by a communications network connected to different hardware and software; (7) facilitating direct commands to the node resources to initiate corrective action; and/or (8) sharing data with other internal and external computer systems.

The present invention will now be described in more detail by illustrative examples with reference to the embodiment(s) depicted in the Figures. The following described embodiment(s) is presented by way of example and should not be construed as limiting the inventive concept to any particular configuration.

Referring to Figure 1, a basic overview of the network topology according to the present invention is depicted. As shown in Figure 1, the network management system

server **1** is coupled to a plurality of nodes **3a-3f**, **4a**, and **4b** via a network **2** using Transmission Control Protocol/Internet Protocol (TCP/IP). TCP/IP is the basic communication language or protocol of world wide communications networks, such as, the Internet. TCP/IP can also be used as a communications protocol in the private networks called intranets and in extranets.

The exception-based system management system includes at least one client terminal **5a**, **5b**. The client terminal **5a**, **5b** typically includes a central processing unit (CPU), a monitor or other visual display device, a keyboard or some other input device, and a communications device, such as a modem. Further, each client terminal **5a**, **5b** is electronically connected to at least one network **2**.

ATMs **3a-3f** transmit and receive data from a network management system server **1** via a network **2**. Bank servers **4a** and **4b** also transmit and receive data from the network management system server **1** via the network **2**. Client terminals **5a-5b** interact with the network management system server **1** in a typical client/server platform.

The operation of the system according to the embodiment shown in Figure 1 is as follows. ATM **3a** has an exception. An exception may be a mechanical problem, such as a jammed door or malfunctioning key, or a software-related problem, such as an error in keeping time of certain transactions. As the exception is raised at ATM **3a**, the exception is transmitted to the network management system server **1** via the network **2**. The network management system server **1** either creates a new object in the software or modifies an existing object to show the exception. A user sitting at a client terminal **5a**, **5b** enters the system and requests information about specific exceptions raised or about

specific nodes within the system.

The user at a client terminal **5a**, **5b** may request information based on the exceptions, geography of various nodes, by business branch, status of delivery, or functionality (e.g., ATM vs. business server). Typically, the exception-based system monitoring system operates mostly on the exceptions raised, as many systems are prohibitively large to view in other formats.

Once the user has received requested information regarding exceptions, the user selects one of the exceptions to process. The user may then either attempt to correct the exception or request more information from the system in order to more completely understand the nature of the exception and its origin. The network management system server **1** provides the requested information to the appropriate client terminal **5a**, **5b** so the user may identify the problem and initiate corrective action. The user views the exception and issues a correction command from a client terminal **5a**, **5b**. The correction is forwarded from the network management system server **1** to the appropriate ATM **3a-3f**. If the correction resolves the exception, a correction acknowledgment signal is transmitted from the ATM **3a-3f** to the network management system server **1** via the network **2**. The network management system server **1** reports both the exception and the correction and then removes the exception from the exception list.

In order to accomplish the above described functionality according to an embodiment of the present invention, the network management system server **1** uses a node configuration data tool to build and design new nodes to be added to the system. In this manner, the network management system server **1** and its corresponding client

terminals **5a-5b**, may accept and process new nodes.

In an embodiment of the present invention, not all users are permitted to access the same correction capabilities. Every user of the network management system server **1** must go through a security check. Users of the network management system server **1** are organized into a security hierarchy. High-level security users are permitted to access more sensitive system functions than low-level security users. Thus, highly secure users may access software to remedy a wide range of system exceptions, while lower security level users may remedy a limited number of exceptions.

A feature of the present invention is the use of universal code, such that any client terminal **5a**, **5b** may access the network management system server **1** regardless of the platform of the client terminal **5a**, **5b**. Thus, a user may initiate a corrective action for an exception at a first client terminal, such as, for example, a PC terminal, be interrupted, and complete the corrective action at a second client terminal, such as, for example, a UNIX terminal.

The network management system server **1** provides the user with the status of all exceptions/exceptions, status of all branches in a business or in multiple businesses, status of all delivery systems in a business branch, and status of a specific delivery system.

In one preferred embodiment, the implementation of the present invention uses a Java applet, which may be accessed over a world wide communications network, such as, for example, the Internet or an Intranet. As will be appreciated by those skilled in the art, the Java language developed by Sun Microsystems of Palo Alto, California, is an example of a language that can be run on a plurality of machines regardless of the platform of the

machine. Thus, in one embodiment of the present invention, only a Java application is required for a user to interact with the exception-based system management system. Similarly, web-based technology (e.g., web browser and web servers) could be used to provide user interaction with the exception-based system management system.

In another embodiment of the present invention, the system monitors and displays the state of one or more managed delivery system nodes. These delivery system nodes include ATMs and their supporting devices, home banking servers, and other terminals and servers that support a financial institution. The present invention also supports remote inquiries and control commands and tracks exceptions from inception to resolution.

Referring now to Figure 2A entitled *Exception-Based System Management System Component Object Model* and Figure 2B entitled *Exception-Based System Management System Component Object Model(2)*, the internal software architecture of the exception-based system management system is illustrated. Except for the two (2) UNIX processes **256**, **258** shown in Figure 2B, all of the boxes contained in these diagrams represent object classes that are implemented in the Java programming language. The present invention is a distributed application wherein a "client" portion runs on any machine with network accessibility to the network management system server system. A corresponding "server" portion runs on a system management server known as the network management system server. Both machines require a Java Runtime Environment (JRE).

Figure 2A illustrate the internal architecture with respect to the "client" portion. The bold box at the top of the diagram represents the network management system server

210. The other components are object classes that implement the "client" side of the distributed application. The labeled arrows show the high level functional interactions between the corresponding objects. The client portion consists of five major object types, including, Dispatcher **212**, INCComm **214**, Activatable **216**, Cache **218**, and Store **220**.

Dispatcher **212** is the controlling component for the entire application. It is responsible for the creation of the application window and orchestrates the creation of the other objects. It is also responsible for the top-level toolbar that controls the main transition between screens.

INCComm **214** is the interface to the network management system server **210** portion of the application. All requests for server data pass through this object. This includes requests for screen specific data, command and command response data, and user specific configuration requests. INCComm **214** translates object-oriented requests into a non-object-oriented format that is understood on the network management system server **210** side by the HyperINC process.

Activatable **216** is the parent class to the many object classes shown beneath. In object-oriented programming terms, this is known as inheritance. Each of the Activatable object classes implements a specific screen. At any one time there is one and only one screen showing in the application window. This is the currently active screen. Transition from the currently active screen to another screen is initiated by the user clicking a button from the main top-level toolbar or by the user clicking a special button inside the currently active screen. Under the control of the Dispatcher **212**, the currently active screen is deactivated and the selected screen is then activated.

Cache **218** is another parent class to the object classes shown beneath. Cache objects are data storage components that require interaction with the network management system server **210**. They represent user specific configuration data such as business registration, filtering criteria, or other configuration data that must be saved on the network management system server **210**. Cache objects are populated with data when the user logs into the exception-based system management system. If the user subsequently requests an update of the configuration data, the Cache object representing that data interacts with the INCComm object to make that happen. The INCComm object then interacts with the network management system server **210** to make the change.

Store **220** is another parent class to the object classes shown beneath. Like the Cache objects, Store objects store data. However, this data is static and never changes during the life of the user session. Each Store **220** is populated at user login by accessing files that reside on the network management system server **210**.

Figure 2B illustrates the internal architecture with respect to the network management system "server" portion. The bold box at the top of the diagram represents the exception-based system management system client **250**. The other components are object classes and UNIX processes that implement the server side of the distributed application. The labeled arrows show the high level functional interactions between the corresponding objects. The server portion consists of object classes INCRemoteFactory **252** and RemoteTCPStream **254**, and UNIX processes HyperINC **256** and TCP/IP Pass-thru-Gateway **258**.

INCRemoteFactory **252** is the starting place for the server side of the application.

There is only one instance and it is responsible for creating and removing individual instances of RemoteTCPStream **254**. When a user at a remote client site attempts to login, the INCRemoteFactory **252** creates a new instance of a RemoteTCPStream **254** and directs the client to further interact with the newly created RemoteTCPStream **254**.

RemoteTCPStream **254** is responsible for translating object-oriented remote method calls into the message-based protocol understood by the HyperINC process **256**. It works in conjunction with a TCP/IP Pass-thru-Gateway **258**, which further translates the message protocol from IP sockets into a proprietary communication transport called Transparent Transport Mechanism (TTM).

The HyperINC process **256** is the main logic component of the server side of the present invention. It interacts with the many other network management system server components that comprise the network management system server system infrastructure. It does this to acquire the data necessary to populate the various user modules.

Turning now to the illustrative user modules depicted in Figures 3 through 10, these modules may be advantageously displayed as a web page using Microsoft Active Server Page technology including scripts written in Visual Basic or Java Script. Java, Javascript, or Active X form may also be used, and provide the added ability to perform validity checks on the attributes as they are input into a client terminal. These trade names are commonly known terminology in the art.

In a preferred embodiment of the present invention, the client terminal may be any PC running a Windows operating system or may be a Windows NT workstation with access to a global communications network, such as, the Internet. For example, the client

terminal may be a PC that supports either Internet Explorer or Navigator to provide access to the Intranet or Internet. Alternatively, it should be appreciated that the client terminal could take on a variety of other suitable forms, such as, for example, PC's and/or servers running UNIX or LINUX, a Macintosh, a personal digital assistant (PDA), or a pen-based computer. Furthermore, the client terminal could be electronically connected to a network communications system by way of other wireline or wireless technology, including, for example, WAN, LAN, satellite system, telephone lines, and the like.

The user modules selectively access software objects populated with data at login time and prompt the user to select and enter various information about an exception, trouble ticket, and associated information. These modules can be categorized into the following three categories: administrative modules, monitoring and display modules, and trouble ticket modules. Administrative modules include the "Login" Module (user identification and password) and the "Administration" Module (filtering, registration, operational parameters). Monitoring and display modules include the "Branch" Module (business branch status and configuration), the "Exception" Module (outstanding network exceptions), the "Detail" Module (exceptions on a particular node), the "Command" Module (issue commands to remote nodes and view responses), and the "Status" Module (high level status of each associated business). Finally, the trouble ticket modules include the "Ticket" Module (trouble ticketing) and the "Ticket Browser" Module (trouble ticket selection and management).

Figure 3 illustrates the "Login" Module that allows a user to enter his or her login identification and password to access the exception-based system management system.

Referring now to Figures 4A-4H, the administrative user modules represent screen shots that display and prompt the user to view, input, select, and/or transmit administrative information. For example, administrative information displayed in Figure 4B represent various business branches that can be selected by checking the registration box. Specifically, a user would use the registration window to identify and to select the various business segments. At the attribute prompt "Registration," a user would single-click the left mouse button to either check or un-check each listed business. After selecting all of the business branches, the user would click on the "Register" icon to transmit the selected business branches to the system. Similar steps would be completed for each of the additional attribute prompts in Figures 4C through 4H.

The administration modules give a user high-level information about the network and allow a user to perform exception-based management administrative tasks. These modules serve as an entry into other modules, which are accessed by clicking on a tab at the top of each module.

The administrative modules consist of numerous modules as described with reference to each of the following figures. Figure 4A illustrates the "Home" Module that allows a user to view session information. Figure 4B illustrates the "Registration" Module that enables a user to select and register one or more businesses. Figure 4C illustrates the "Alertable Filtering" Module that enables a user to configure the system to automatically go into a special state whenever a predetermined period of user inactivity is detected. In this state, the present invention can audibly alert a user of the presence of any exception events that a business deems worthy of notification. Figure 4D illustrates the

"Device Filtering" Module that enables a user to eliminate messages from managed nodes or specific devices on the nodes. Figure 4E illustrates the "Message Filtering" Module that enables a user to eliminate particular exception events from consideration. Figure 4F illustrates the "Node Filtering" Module that enables a user to eliminate the display of exception messages from specific nodes. Figure 4G illustrates the "Runtime Control" Module that enables a user to choose the type of GUI and to customize other features, such as whether to display international date format. Figure 4H illustrates the "Documentation" Module that enables a user to view the on-line user's guide for help and examples.

Figure 5 illustrates the "Branch" Module that provides an overview of all the businesses for which a user is registered. It also lets a user see details of a business branch and an indication of a node exception within the business branch. Specifically, a user can view information about the registered network management system server domain, a specific business, and a branch within a business. From this module, a user can directly access the associated "Detail" Module to see specific exception detail information about a node.

Figure 6 illustrates the 'Exceptions" Module, which is the focal point for determining which actions to take to resolve a node exception. It shows each node that has an outstanding exception, the most severe exception with that node, whether or not all exceptions with that node have received user attention, and the identification of the user attending to the exception. The present invention prioritizes the severity of an exception using a set of rules that is programmed into the network management system server. To

get detailed information about the exception, a user can directly access the associated "Detail" Module. From this module, a user can also access the "Branch" Module by clicking on the name of the exception.

Figure 7 illustrates the "Detail" Module that provides information about specific exceptions with specific nodes and lets the user take corrective action on the particular exception as well as view detailed information about the exception. A "Detail" Module is available for each node and contains the full message text for each exception. For each type of exception, a suggested resolution can be displayed by clicking on the exception icon. The "Detail" Module enables a user to directly access the "Branch" Module to highlight and view a particular node in the business or business segment. The "Detail" Module enables a user to directly access the "Command" Module and issue a command to a particular node.

In addition, the "Detail" Module screen provides icons that allow for management of individual exceptions. This is accomplished by selecting an exception from the exception list and clicking to acknowledge, create a trouble ticket, or remove the exception from the system.

Figure 8 illustrates the "Command" Module that enables a user to inquire about the status of a node, its controller (if appropriate), its connection to the network management system server or front end processor (FEP), and the current version of software running on the node. In addition, the "Command" Module lets a user directly issue commands to a node to start, stop, reboot, and change the current version of software. Inquiries and commands are sent from the present invention to the network management system server.

The network management system server forwards the inquiries and commands to the appropriate node and returns the responses. Furthermore, a user can view outstanding commands and responses that have been routed back to the network using the "Command" Module.

Figures 9A and 9B represent the "Ticket" and "Ticket Search" Modules. These modules allow a user to create a report, known as a "trouble ticket," that describes an exception in detail and tracks that exception through to its resolution. These module enable a user to perform the following functions: (1) create, search, and print trouble tickets; (2) view configuration information, exception descriptions, user identifications, and automatically include time stamps; (3) add comments to a trouble ticket (previous comments are preserved so there is a history of the exception resolution); (4) close trouble tickets automatically or manually; (5) review both open and closed trouble tickets, wherein an authorized user can view, update, or delete trouble tickets; and (6) list trouble tickets by individual user identification. Additionally, a user can browse all trouble tickets created by the present invention (except those deleted) and select multiple criteria to display the trouble ticket, such as, for example, by a specific business, business branch, node, start or end date, exception code or type, login identification, and so on.

Figure 10 illustrates the "Status" Module that provides the status of all monitored nodes within a business. It shows high-level statistics for both the ATMs and the other servers, such as, for example, bank servers and other terminals and servers within a financial institution. For each business, the module has a tab labeled with the business name. The module displays the ATM and server information separately. The "Status"

Module presents the information in horizontal bar-graph form, divided into nodes that are "OK," "Degraded," "Down," or "Never Connected." It gives both percentages and actual counts.

The term "OK" indicates that there are no exceptions within the node. "Degraded" indicates that the there are one or more exceptions within the node, but the node still has partial functions. "Down" indicates that the node is not functioning. "Never Heard From" indicates that the node has never communicated with the network management system server.

Furthermore, a change in the status of a part of the network is indicated by a message that a managed node sends to the network management system server. The network management system server takes these messages and updates its databases with the latest information about the node. The invention accesses the database and displays information in all of the status display modules. For example, the types of messages transmitted from the ATMs to the network management system server that indicate status are listed in the table below:

Status examples transmitted from the ATMs to the network management system server
Devices down (terminal, cash dispenser, depository, etc.)
Devices up (terminal, cash dispenser, depository, etc.)
Replies to inquiry commands issued from either the network management system server or any present invention terminal
Warnings (e.g., "cash low" and "cash very low")
"Safe open" and "Safe closed"
Network management system server connect and disconnect

The foregoing description and associated figures detail only illustrative examples

of the environment in which the invention can be used and are not intended to be limiting. For instance, the user modules shown in Figures 3 through 10 are by way of example only. In addition, attributes can be constantly updated and additional fields can be added by authorized users and/or authorized businesses. Furthermore, the programming languages, software platforms, operating systems, and web-based technology mentioned in the description are by way of example only, and the present invention may always be enhanced to incorporate the most advanced available technology. From the teaching of the present description, the person skilled in the art will be able to implement the invention with different embodiments.

GLOSSARY OF ABBREVIATIONS

This disclosure, including illustrative information used to populate user modules in the figures, makes use of certain abbreviations which have the following meanings:

Applet	Short-term Java application that is automatically downloaded via a web page
ATM	Automated Teller Machine
CAT	Customer Activated Terminal
CDM	Cash-dispensing Mechanism
CIU	Customer Interface Unit (the graphics & touch module controller portion of a CAT)
CMTTS	Citibanking Management and Trouble Tracking Subsystem
CPU	Central Processing Unit
CS	Citishield Device
CUI	Command User Interface
DEP	Depositor
FEP	Front End Processor
FLS	Flash Device
GUI	Graphical User Interface
HST	Host
INC	Integrated Network Control
IP	Internet Protocol
JRE	Java Runtime Environment
LAN	Local Area Network
PBX	Private Branch Exchange
PD	Proximity Detector
PIN	Personal Identification Number Pad Device
PTR	Printer
RDR	Card Reader
RMI	Remote Method Invocation
SAF	Safe Device
SCM	SCM Device
SW	CIU Panel Tamper Switch
TCP/IP	Transmission Control Protocol/Internet Protocol
TRD	Transport Reader
TRM	Terminal
TTM	Transparent Transport Mechanism
WAN	Wide Area Network